Прикладная линейная алгебра

Домашняя контрольная работа 2

Поздняков Виталий, Вариант 18

---

1. Найти лучшее приближение матрицы A_1 ранга 2 матрицы A в норме $\|\cdot\|_2$ и найти $\|A - A_1\|_2$ где A

In [1]:

```
import numpy as np
```

In [2]:

```
A = np.array([[10, -4, -7, -1],
              [15,  25, 8, -4],
              [12,  1,  2, -5],
              [-8, -6,  0, -4]])
```

In [3]:

```
np.linalg.matrix_rank(A)
```

Out[3]:

```
4
```

Ранг матрицы $rk(A) = 4$, значит сама матрица A не может быть использована в качестве приближения.

По теореме приближения матриц по унитарным инвариантным нормам лучшее приближение матрицы A есть матрица A_1 такая что

$$A_1 = U^*\Sigma^0_{(r)}V \approx A$$

Где $U^*\Sigma^0_{(r)}V$ — сингулярное разложение, в котором матрица Σ имеет все нулевые сингулярные значения старше индекса r. Найдем сингулярное разложение.

In [4]:

```
U, Sigma, V = np.linalg.svd(A)
print('U:')
print(U.round(2))
print('Sigma:')
print(np.diag(Sigma).round(2))
print('V:')
print(V.round(2))
```

```
U:
[[-0.05  0.8  -0.25 -0.55]
 [-0.92 -0.25  0.04 -0.3 ]
 [-0.28  0.53  0.63  0.5 ]
 [ 0.27 -0.17  0.73 -0.6 ]]
Sigma:
[[32.81  0.    0.    0.  ]
 [ 0.   15.77  0.    0.  ]
 [ 0.    0.    7.09  0.  ]
 [ 0.    0.    0.    3.28]]
V:
[[-0.6  -0.75 -0.23  0.12]
 [ 0.76 -0.5  -0.41 -0.11]
 [-0.04 -0.25  0.47 -0.84]
 [ 0.25 -0.35  0.74  0.51]]
```

Теперь заменим все сингулярные значения старше второго на ноль

In [5]:

```
Sigma1 = np.diag(np.array([Sigma[0], Sigma[1], 0, 0]))
```

Тогда матрица A_1 имеет вид

In [6]:

```
A1 = (U @ Sigma1 @ V)
print(A1.round(2))
```

```
[[10.39 -5.08 -4.81 -1.6 ]
 [15.26 24.73  8.6  -3.28]
 [11.75  2.7  -1.33 -2.06]
 [-7.31 -5.37 -0.97  1.38]]
```

Проверим ранг матрицы A_1

In [7]:

```
np.linalg.matrix_rank(A1)
```

Out[7]:

```
2
```

Значит полученная матрица A_1 – ближайшее приближение ранга 2 матрицы A.

Теперь найдем значение нормы разности этих матриц. Пусть $\Delta A = A - A_1$

In [8]:

```
deltaA = A - A1
print(deltaA.round(2))
```

```
[[-0.39  1.08 -2.19  0.6 ]
 [-0.26  0.27 -0.6  -0.72]
 [ 0.25 -1.7   3.33 -2.94]
 [-0.69 -0.63  0.97 -5.38]]
```

По свойству спектральной нормы $\|\Delta A\|_2 = \sigma_1$, то есть спектральная норма равна максимальному сингулярному значению. Найдем сингулярное разложение матрицы ΔA.

In [9]:

```
U, Sigma, V = np.linalg.svd(deltaA)
print('U:')
print(U.round(2))
print('Sigma:')
print(np.diag(Sigma).round(2))
print('V:')
print(V.round(2))
```

```
U:
[[ 0.25  0.55 -0.54  0.59]
 [-0.04  0.3  -0.56 -0.77]
 [-0.63 -0.5  -0.55  0.24]
 [-0.73  0.6   0.31  0.04]]
Sigma:
[[7.09 0.   0.   0.  ]
 [0.   3.28 0.   0.  ]
 [0.   0.   0.   0.  ]
 [0.   0.   0.   0.  ]]
V:
[[ 0.04  0.25 -0.47  0.84]
 [-0.25  0.35 -0.74 -0.51]
 [-0.72 -0.66 -0.16  0.14]
 [ 0.65 -0.61 -0.44 -0.09]]
```

Таким образом $\|A - A_1\|_2 = 7.09$

2. Оцените с помощью числа обсуловленности матрицы A относительную погрешность приближенного решения $(1, 1)$ системы $AX = B$ в норме $|\cdot|_1$, где A и B равны соответсвенно

In [10]:

```
A = np.array([[0.02,  0.4],
              [1.03, -1.07]])

B = np.array([[-7.05],
              [-5.1]])
```

Приближенное решение $\hat{X} = (1, 1)^T$

Обозначим относительную погрешность δX

По свойству числа обусловленности:

$$\frac{1}{\varkappa(A)}\delta B \leq \delta X \leq \varkappa(A)\delta B$$

Число обусловленности матрицы A по определению

In [11]:

```
kappaA = np.linalg.norm(A, 1) * np.linalg.norm(np.linalg.inv(A), 1)
round(kappaA, 2)
```

Out[11]:

```
7.12
```

$\varkappa(A) = \|A\|\|A^{-1}\| = 7.12$

Возьмем произвольный приближенный вектор $\hat{B}$, например

In [12]:

```
hatB = B.round()
print(hatB)
```

```
[[-7.]
 [-5.]]
```

Теперь вычислим $\delta B = \frac{|\Delta B|_1}{|B|}$

In [13]:

```
deltaB = np.linalg.norm(B - hatB, 2) / np.linalg.norm(B, 2)
print(round(deltaB, 2))
```

```
0.01
```

Тогда нижняя граница

In [14]:

```
round(1/kappaA * deltaB, 4)
```

Out[14]:

```
0.0018
```

Верхняя граница

In [15]:

```
round(kappaA * deltaB, 4)
```

Out[15]:

```
0.0915
```

Таким образом, $0.001 \leq \delta x \leq 0.092$

3. Решить приближенно систему и оценить погрешности решения в нормах $|\cdot|_1, |\cdot|_2, |\cdot|_\infty$:

$$\begin{cases} 0(0+\epsilon_1)x + 2(-2+\epsilon_2)y = 3+\epsilon_3 \\ -1x + (-3+\epsilon_1)y = 1+\epsilon_4 \end{cases}$$

где неизвестные числа ϵ_j удовлетворяют условиям $|\epsilon_j| < 0.05$ для всех j

Сначала решим систему символьно:

$$\begin{cases} 0(0+\epsilon_1)x + 2(-2+\epsilon_2)y = 3+\epsilon_3 \\ -1x + (-3+\epsilon_1)y = 1+\epsilon_4 \end{cases}$$

$$= \begin{cases} y = \frac{3+\epsilon_3}{(-4+2\epsilon_2)} \\ -x + (-3+\epsilon_1)y = 1+\epsilon_4 \end{cases} \begin{cases} y = \frac{3+\epsilon_3}{(-4+2\epsilon_2)} \\ x = -1 - \epsilon_4 + (-3+\epsilon_1)\frac{3+\epsilon_3}{(-4+2\epsilon_2)} \end{cases}$$

Представим систему в векторном виде $AX = B \Leftrightarrow \begin{pmatrix} 0 & -4+2\epsilon_2 \\ -1 & -3+\epsilon_1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 3+\epsilon_3 \\ 1+\epsilon_4 \end{pmatrix}$

Теперь оценим решение по свойству числа обусловленности

$$\frac{1}{\varkappa(A)}(\delta B + \delta A) \leq \delta X \leq \varkappa(A)(\delta B + \delta A)$$

Так как ϵ_j неизвестны, то будем оценивать решение по максимально широкому интервалу. Найдем значения ϵ_1 и ϵ_2 при которых $\varkappa(A)$ максимально. Без потери общности, по норме $|\cdot|_1$:

In [16]:

```
eps = [-0.05, 0.05, 0, 0]
hatA = np.array([[0,  -4 ],
                 [-1, -3 ]])
A = np.array([[0,  -4 + 2*eps[1]],
              [-1, -3 +   eps[0]]])
kappaA = np.linalg.norm(A, 1) * np.linalg.norm(np.linalg.inv(A), 1)
round(kappaA, 2)
```

Out[16]:

```
7.22
```

Значит $\epsilon_1 = -0.05$ и $\epsilon_2 = 0.05$

Аналогичным образом найдем ϵ_3 и ϵ_4 при которых δB максимально.

In [17]:

```
eps = [-0.05, 0.05, -0.05, -0.05]
hatB = np.array([3, 1])
B = np.array([3 + eps[2], 1 + eps[3]])
deltaB = np.linalg.norm(B - hatB, 1) / np.linalg.norm(B, 1)
print(round(deltaB, 3))
```

```
0.026
```

Значит $\epsilon_3 = -0.05$ и $\epsilon_4 = -0.05$

Теперь оценим относительную погрешноть решения. Нижние границы:

In [18]:

```
norms = [1, 2, np.inf]
for n in norms:
    kappaA = np.linalg.norm(A, n) * np.linalg.norm(np.linalg.inv(A), n)
    deltaA = np.linalg.norm(A - hatA, n) / np.linalg.norm(A, n)
    deltaB = np.linalg.norm(B - hatB, n) / np.linalg.norm(B, n)
    print('По норме {}:'.format(n), round(1/kappaA * (deltaB + deltaA), 4))
```

```
По норме 1: 0.0065
По норме 2: 0.0071
По норме inf: 0.0058
```

Верхние границы:

In [19]:

```
norms = [1, 2, np.inf]
for n in norms:
    kappaA = np.linalg.norm(A, n) * np.linalg.norm(np.linalg.inv(A), n)
    deltaA = np.linalg.norm(A - hatA, n) / np.linalg.norm(A, n)
    deltaB = np.linalg.norm(B - hatB, n) / np.linalg.norm(B, n)
    print('По норме {}:'.format(n), round(kappaA * (deltaB + deltaA), 4))
```

```
По норме 1: 0.3408
По норме 2: 0.2887
По норме inf: 0.3005
```

4. Найдите приближенно обратную матрицу к матрице A и оцените погрешность приближения относительно равномерной нормы если элементы матрицы A известны с абсолютной погрешностью 0.01

In [20]:

```
A = np.array([[ 5, -7],
              [-1,  1]])

Delta = np.array([[0.01, 0.01],
                  [0.01, 0.01]])
```

Оценим относительную погрешность сверху

$$\delta A^{-1} \le \varkappa(A)\frac{\|\Delta\|}{\|A\|}$$

In [21]:

```
round(np.linalg.norm(A, np.inf)
      * np.linalg.norm(np.linalg.inv(A), np.inf)
      * (np.linalg.norm(Delta, np.inf)
         / np.linalg.norm(A, np.inf)), 2)
```

Out[21]:

```
0.08
```

Матрица A^{-1} имеет вид

In [22]:

```
print(np.linalg.inv(A))
```

```
[[-0.5 -3.5]
 [-0.5 -2.5]]
```

5. Решите систему линейный уравнений методом итераций

$$\begin{cases} 24x + 8y + 5z = 7 \\ 6x + 23y + 8z = 5 \\ 1x + 1y + 20z = 7 \end{cases}$$

Определите номер итерации, после которой погрешность приближения по каждой координате не превосходит 0.01 и найдите соответствующее приближенное решение

In [23]:

```
A = np.array([[24, 8, 5],
              [6, 23, 8],
              [1, 1, 20]])

B = np.array([7, 5, 7])
```

Сначала приведем систему к виду, удобному для итераций

$$X_{k+1} = PX_k + B$$

Учитывая диагональное преобладание матрицы A

$$\begin{cases} 24x + 8y + 5z = 7 \\ 6x + 23y + 8z = 5 \\ 1x + 1y + 20z = 7 \end{cases} \Leftrightarrow \begin{cases} 24x = -8y - 5z + 7 \\ 23y = -6x - 8z + 5 \\ 20z = -1x - 1y + 7 \end{cases} \Leftrightarrow \begin{cases} x = -8y/24 - 5z/24 + 7/24 \\ y = -6x/23 - 8z/23 + 5/23 \\ z = -1x/20 - 1y/20 + 7/20 \end{cases}$$

Тогда матрица P и вектор B имеют вид

In [24]:

```
P = np.array([[0,      -8/24, -5/24],
              [-6/23,  0,     -8/23],
              [-1/20, -1/20,  0    ]])

B = np.array([7/24, 5/23, 7/20])
```

In [25]:

```
round(np.linalg.norm(P, 1), 2)
```

Out[25]:

```
0.56
```

$\|P\|_1 < 1$ значит система сходится к решению

По теореме сходимости итеративных методов для достижения точности ϵ требуется N итераций.

$$N = \log_{\|P\|}\left(\frac{\epsilon(1 - \|P\|)}{|B|}\right) - 1$$

Тогда для точности 0.01:

In [26]:

```
import math
normP = np.linalg.norm(P, 1)
normB = np.linalg.norm(B, 1)
round(math.log((0.01 * (1 - normP))/ normB, normP) - 1, 2)
```

Out[26]:

```
7.97
```

Потребутся 8 итераций

In [27]:

```
X = np.array([0, 0, 0])
for i in range(8):
    X = P@X + B
    print(i+1, X.round(4))
```

```
1 [0.2917 0.2174 0.35  ]
2 [0.1463 0.0196 0.3245]
3 [0.2175 0.0663 0.3417]
4 [0.1984 0.0418 0.3358]
5 [0.2078 0.0488 0.338 ]
6 [0.205  0.0456 0.3372]
7 [0.2062 0.0466 0.3375]
8 [0.2058 0.0462 0.3374]
```

6. Найдите самую влиятельную вершину в графе с помощью алгоритма PageRank, где матрица смежноси графа

In [28]:

```
A = np.array([[1, 1, 1, 1],
              [1, 1, 1, 1],
              [1, 0, 0, 1],
              [0, 1, 0, 1]])
```

Составим матрицу перехода P

In [29]:

```
P = np.array([A[0]/4, A[1]/4, A[2]/2, A[3]/2]).T
print(P)
```

```
[[0.25 0.25 0.5  0.  ]
 [0.25 0.25 0.   0.5 ]
 [0.25 0.25 0.   0.  ]
 [0.25 0.25 0.5  0.5 ]]
```

Решение будем искать в виде $X = MX$, где $M = (1 - \beta)P + \beta Q$

Пусть $\beta = 0.15$

In [30]:

```
beta = 0.15
```

По определению матрица Q имеет вид

In [31]:

```
Q = np.ones([4, 4])/4
print(Q)
```

```
[[0.25 0.25 0.25 0.25]
 [0.25 0.25 0.25 0.25]
 [0.25 0.25 0.25 0.25]
 [0.25 0.25 0.25 0.25]]
```

Тогда решение

In [32]:

```
M = (1 - beta)*P + beta*Q
X = np.ones(4)/4
for i in range(8):
    X = M@X
    print(X.round(2))
```

```
[0.25 0.25 0.14 0.36]
[0.2  0.3  0.14 0.36]
[0.2  0.3  0.14 0.36]
[0.2  0.3  0.14 0.36]
[0.2  0.3  0.14 0.36]
[0.2  0.3  0.14 0.36]
[0.2  0.3  0.14 0.36]
[0.2  0.3  0.14 0.36]
```

Таким образом, 4 вершина самая влиятельная